(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号
特開2003－198383
(P2003－198383A)
(43)公開日　平成15年７月11日(2003.7.11)

(51)Int.Cl.⁷	識別記号	FI		テーマコード(参考)
H03M 13/19		H03M 13/19		5B001
G06F 11/10	330	G06F 11/10	330Q	5J065
H04L 1/00		H04L 1/00	A	5K014

審査請求　未請求　請求項の数７　ＯＬ　（全 15 頁）

(21)出願番号　特願2001－397922(P2001－397922)

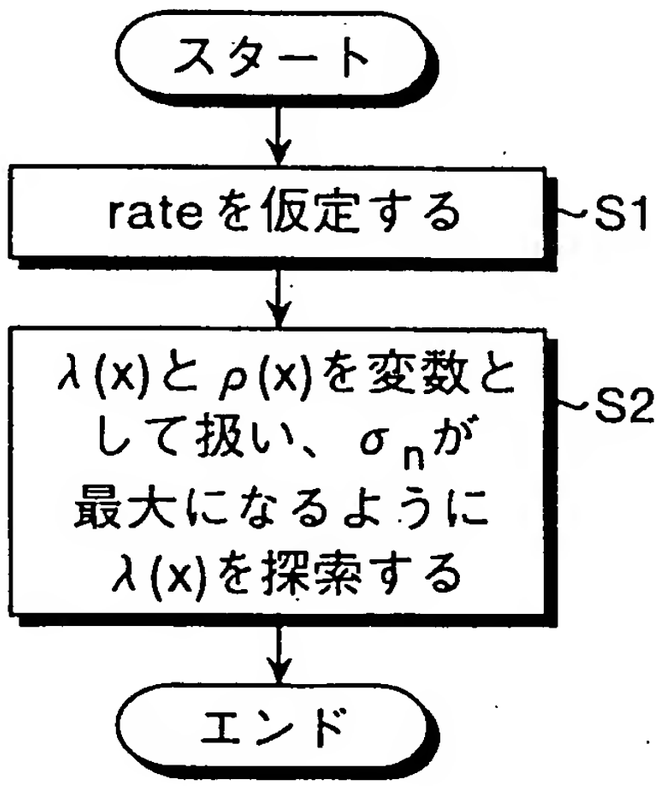
(22)出願日　平成13年12月27日(2001.12.27)

(71)出願人　000006013
三菱電機株式会社
東京都千代田区丸の内二丁目２番３号
(72)発明者　松本　渉
東京都千代田区丸の内二丁目２番３号　三菱電機株式会社内
(74)代理人　100089118
弁理士　酒井　宏明

Ｆターム(参考)　5B001 AA01 AB03
5J065 AA01 AB01 AC02 AF02 AH01 AH15
5K014 AA01 BA02 GA05

(54)【発明の名称】　ＬＤＰＣ符号用検査行列生成方法

(57)【要約】
【課題】　確定的でかつ特性が安定したＬＤＰＣ符号用の検査行列を、短時間で容易に探索可能なＬＤＰＣ符号用検査行列生成方法を得ること。
【解決手段】　本発明のＬＤＰＣ符号用検査行列生成方法は、復号器における入出力データの対数尤度比がガウス分布に近似できると仮定してＬＤＰＣ符号の「Ｓｕｍ－Ｐｒｏｄｕｃｔアルゴリズム」を解析し、符号化レートを固定した状態で、かつガウスノイズが最大になるように、行の重みと列の重みの最適なアンサンブルを１回の線形計画法で探索し、当該アンサンブルにしたがってＬＤＰＣ符号用の検査行列を生成する。



1

【特許請求の範囲】

【請求項１】　復号器における入出力データの対数尤度比がガウス分布に近似できると仮定してＬＤＰＣ（Low-Density Parity-Check）符号の「Ｓｕｍ－Ｐｒｏｄｕｃｔアルゴリズム」を解析することによって、誤りが０となるＳＮＲの限界（threshold）を求めるＬＤＰＣ符号用検査行列生成方法において、
符号化レートを固定した状態で、かつガウスノイズが最大になるように、行の重みと列の重みの最適なアンサンブル（thresholdが最小となるアンサンブル）を１回の線形計画法で探索し、当該アンサンブルにしたがってＬＤＰＣ符号用の検査行列を生成することを特徴とするＬＤＰＣ符号用検査行列生成方法。

【請求項２】　前記アンサンブルを探索後、当該探索結果に基づいてユークリット幾何符号の各行または各列からランダムに「１」を抽出し、各行または各列を分割することによって、Ｉｒｒｅｇｕｌａｒ－ＬＤＰＣ符号の検査行列を生成することを特徴とする請求項１に記載のＬＤＰＣ符号用検査行列生成方法。

【請求項３】　前記アンサンブルの重み配分を、重み単位の重み総数が整数で、かつ重み単位の重み総数の総和とユークリット幾何符号の「１」の総数とが等しくなるように調整し、調整後のアンサンブルに基づいて分割処理を行うことを特徴とする請求項２に記載のＬＤＰＣ符号用検査行列生成方法。

【請求項４】　基本のランダム系列のラテン方陣を作成し、
ユークリッド幾何符号の第ｍ列目をｎ分割する場合、上記ラテン方陣の第ｍ行目のランダム系列をｎ分割し、当該ｎ分割後の各ランダム系列を用いてユークリッド幾何符号の第ｍ列目の「１」を抽出することを特徴とする請求項２または３に記載のＬＤＰＣ符号用検査行列生成方法。

【請求項５】　基本のランダム系列のラテン方陣を作成し、
ユークリッド幾何符号の第ｍ行目をｎ分割する場合、上記ラテン方陣の第ｍ行目のランダム系列をｎ分割し、当該ｎ分割後の各ランダム系列を用いてユークリッド幾何符号の第ｍ行目の「１」を抽出することを特徴とする請求項２または３に記載のＬＤＰＣ符号用検査行列生成方法。

【請求項６】　基本のランダム系列のラテン方陣を複数個作成し、
列方向に連結したラテン方陣群行列を用いて、ユークリッド幾何符号の第ｍ列目をｎ分割する場合、上記ラテン方陣群行列の第ｍ列目のランダム系列をｎ分割し、当該ｎ分割後の各ランダム系列を用いてユークリッド幾何符号の第ｍ列目の「１」を抽出することを特徴とする請求項２または３に記載のＬＤＰＣ符号用検査行列生成方法。

2

【請求項７】　基本のランダム系列のラテン方陣を複数個作成し、
列方向に連結したラテン方陣群行列を用いて、ユークリッド幾何符号の第ｍ行目をｎ分割する場合、上記ラテン方陣群行列の第ｍ列目のランダム系列をｎ分割し、当該ｎ分割後の各ランダム系列を用いてユークリッド幾何符号の第ｍ行目の「１」を抽出することを特徴とする請求項２または３に記載のＬＤＰＣ符号用検査行列生成方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、誤り訂正符号としてＬＤＰＣ（Low-Density Parity-Check）符号を採用した符号化器におけるＬＤＰＣ符号用検査行列生成方法に関するものである。

【０００２】

【従来の技術】図１３は、ＬＤＰＣ符号化／復号システムを示す図である。図１３において、１０１は符号化器であり、１０２は変調器であり、１０３は通信路であり、１０４は復調器であり、１０５は復号器である。ここでは、従来のＬＤＰＣ符号用検査行列生成方法を説明する前に、ＬＤＰＣ符号を使用した場合の符号化，復号の流れについて説明する。

【０００３】まず、送信側の符号化器１０１では、後述する所定の方法で検査行列Ｈを生成する。そして、以下の条件に基づいて生成行列Ｇを求める。
Ｇ：ｋ×ｎ行列（ｋ：情報長，ｎ：符号語長）
$GH^T=0$（Ｔは転置行列）

【０００４】その後、符号化器１０１では、情報長ｋのメッセージ（$m_1 m_2 \cdots m_k$）を受け取り、上記生成行列Ｇを用いて符号語Ｃを生成する。
$C=(m_1 m_2 \cdots m_k)G$
$=(c_1 c_2 \cdots c_n)$　　（ただし、$H(c_1 c_2 \cdots c_n)^T=0$）

【０００５】そして、変調器１０２では、生成した符号語Ｃに対して、ＢＰＳＫ，ＱＰＳＫ，多値ＱＡＭなどのデジタル変調を行い、送信する。

【０００６】一方、受信側では、復調器１０４が、通信路１０３を介して受け取った変調信号に対して、ＢＰＳＫ，ＱＰＳＫ，多値ＱＡＭなどのデジタル復調を行い、さらに、復号器１０５が、ＬＤＰＣ符号化された復調結果に対して、「ｓｕｍ－ｐｒｏｄｕｃｔアルゴリズム」によるくり返し復号を実施し、推定結果（もとの$m_1 m_2 \cdots m_k$に対応）を出力する。

【０００７】以下、従来のＬＤＰＣ符号用検査行列生成方法について説明する。ＬＤＰＣ符号用の検査行列としては、たとえば、ＬＤＰＣの発案者Ｇａｌｌａｇｅｒにより以下のような行列が提案されている（図１４参照）。

【０００８】図１４に示す行列は、「１」と「０」の２

値の行列で、「１」の部分を塗りつぶしている。他の部分は全て「０」である。この行列は、１行の「１」の数（これを行の重みと表現する）が４で、１列の「１」の数（これを列の重みと表現する）が３であり、全ての列と行の重みが均一なため、これを一般に「Ｒｅｇｕｌａｒ－ＬＤＰＣ符号」と呼んでいる。また、Ｇａｌｌａｇｅｒの符号では、たとえば、図１４に示すように、行列を３ブロックに分け、２ブロック目と３ブロック目に対してランダム置換を行っている。

【０００９】しかしながら、このランダム置換には、所定のルールが無いため、より特性の良好な符号を見つけるためには、計算機による時間のかかる探索を行わなければならなかった。

【００１０】そこで、たとえば、計算機探索によらなくても確定的に行列を生成でき、比較的安定した良好な特性を示すＬＤＰＣ符号として、ユークリット幾何符号を用いる方法が、Ｙ．Ｋｏｕ等（Y. Kou, S. Lin, and M. P. C. Fossorier, "Low Density Parity Check Codes Based on Finite Geometries: A Rediscovery," ISIT2000, pp. 200, Sorrento, Itary, June 25-30, 2000.）によって提案された。この方法では、規則的なｅｎｓｅｍｂｌｅ（アンサンブル）で構成された「Ｒｅｇｕｌａｒ－ＬＤＰＣ符号」について説明されている。

【００１１】ここでは、有限幾何符号の一種であるユークリッド幾何符号ＥＧ（２，2^6）を用いてＬＤＰＣ符号の検査行列を生成する方法について提案され、誤り率10^{-4}点において、シャノン限界から１．４５ｄＢに接近した特性を得ている。図１５は、たとえば、ユークリッド幾何符号ＥＧ（２，2^2）の構成を示す図であり、行，列のそれぞれの重みが４，４の「Ｒｅｇｕｌａｒ－ＬＤＰＣ符号」構造をしている。

【００１２】したがって、ユークリッド幾何符号ＥＧ（ｍ，2^s）の場合、その特性は、以下のように規定される。

符号長：　$n = 2^{2s} - 1$

冗長ビット長：　$n - k = 3^s - 1$

情報長：　$k = 2^{2s} - 3^s$

最小距離：　$d_{min} = 2^s + 1$

密度：　$r = 2^s / (2^{2s} - 1)$

【００１３】図１５を見ても分かるように、ユークリッド幾何符号は、各行の「１」の配置が行毎に巡回シフトした構造になっており、符号が容易にかつ確定的に構成できる特長がある。

【００１４】Ｙ．Ｋｏｕらによる検査行列の生成方法では、さらに、上記ユークリット幾何符号に基づいて行と列の重みを変更し、行，列を必要に応じて拡張している。たとえば、ＥＧ（２，2^2）の列の重みを１／２に分離する場合、Ｙ．Ｋｏｕらの論文では、１列内に４つある重みを１つ置きに２個づつ分離する。図１６は、列の重みを４から２に規則的に分離した例を示す図である。

【００１５】一方、上記「Ｒｅｇｕｌａｒ－ＬＤＰＣ符号」の特性よりも「Ｉｒｒｅｇｕｌａｒ－ＬＤＰＣ符号」の特性の方が良好であることが、Ｌｕｄｙ等（M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Improved Low-Density Parity-Check Codes Using Irregular Graphs and Belief Propagation," Proceedings of 1998 IEEE International Symposium on Information Theory, pp. 171, Cambridge, Mass., August 16-21, 1998.）により報告された。なお、上記「Ｉｒｒｅｇｕｌａｒ－ＬＤＰＣ符号」は、列と行の重みがそれぞれあるいはどちらか一方が均一でないＬＤＰＣ符号を表す。

【００１６】そして、それは、Ｒｉｃｈａｒｄｓｏｎ等（T. J. Richardson and R. Urbanke, "The capacity of low-density parity-check codes under message-passingdecoding," IEEE Trans. Inform. Theory, vol.47, No.2, pp.599-618, Feb. 2001.）、あるいはＣｈｕｎｇ等（S.-Y. Chung, T. J. Richardson, and R. Urbanke, "Analysis of Sum-Product Decoding of Low-Density Parity-Check CodesUsing a Gaussian Approximation," IEEE Trans. Inform. Theory, vol.47, No.2, pp.657-670, Feb. 2001.）によって理論的に解析された。

【００１７】特に、Ｃｈｕｎｇ等は、繰り返し復号器における入力と出力の対数尤度比（ＬＬＲ）がガウス分布に近似できると仮定してＬＤＰＣ符号の「Ｓｕｍ－Ｐｒｏｄｕｃｔアルゴリズム」を解析し、良好な行と列の重みのアンサンブルを求めている。

【００１８】

【発明が解決しようとする課題】しかしながら、たとえば、上記Ｃｈｕｎｇ等による従来のＬＤＰＣ符号用検査行列生成方法は、行内の「１」の点の数（後述するバリアブルノードの次数配分に相当）と、列内の「１」の点の数（後述するチェックノードの次数配分に相当）と、の両方を変数として、下記の（１）式（ｒａｔｅ：符号化率）が最大となるバリアブルノードの次数配分およびチェックノードの次数配分を求めている。すなわち、ＳＮＲ（Signal to Noise Ratio）が最小となるアンサンブルを線形計画法により探索している。

【００１９】

【数１】

$$rate = 1 - \frac{\int_0^1 \rho(x)}{\int_0^1 \lambda(x)} \qquad \cdots(1)$$

【００２０】そのため、上記「ｒａｔｅ」の最大値により得られる検査行列が流動的になり、特性が安定しない、という問題があった。また、従来のＬＤＰＣ符号用検査行列生成方法は、バリアブルノードの次数配分の導

出とチェックノードの次数配分の導出とを所定回数にわたって繰り返し行っているため、探索処理にある程度の時間を要する、という問題もあった。

【００２１】本発明は、上記に鑑みてなされたものであって、確定的でかつ特性が安定したＬＤＰＣ符号用の検査行列を、短時間で容易に探索可能なＬＤＰＣ符号用検査行列生成方法を得ることを目的とする。

【００２２】

【課題を解決するための手段】上述した課題を解決し、目的を達成するために、本発明にかかるＬＤＰＣ符号用検査行列生成方法にあっては、復号器における入出力データの対数尤度比がガウス分布に近似できると仮定してＬＤＰＣ符号の「Ｓｕｍ－Ｐｒｏｄｕｃｔアルゴリズム」を解析することによって、誤りが０となるＳＮＲの限界（threshold）を求めることとし、さらに、符号化レートを固定した状態で、かつガウスノイズが最大になるように、行の重みと列の重みの最適なアンサンブル（thresholdが最小となるアンサンブル）を１回の線形計画法で探索し、当該アンサンブルにしたがってＬＤＰＣ符号用の検査行列を生成することを特徴とする。

【００２３】つぎの発明にかかるＬＤＰＣ符号用検査行列生成方法にあっては、前記アンサンブルを探索後、当該探索結果に基づいてユークリット幾何符号の各行または各列からランダムに「１」を抽出し、各行または各列を分割することによって、Ｉｒｒｅｇｕｌａｒ－ＬＤＰＣ符号の検査行列を生成することを特徴とする。

【００２４】つぎの発明にかかるＬＤＰＣ符号用検査行列生成方法にあっては、前記アンサンブルの重み配分を、重み単位の重み総数が整数で、かつ重み単位の重み総数の総和とユークリット幾何符号の「１」の総数とが等しくなるように調整し、調整後のアンサンブルに基づいて分割処理を行うことを特徴とする。

【００２５】つぎの発明にかかるＬＤＰＣ符号用検査行列生成方法にあっては、基本のランダム系列のラテン方陣を作成し、ユークリッド幾何符号の第ｍ列目をｎ分割する場合、上記ラテン方陣の第ｍ行目のランダム系列をｎ分割し、当該ｎ分割後の各ランダム系列を用いてユークリッド幾何符号の第ｍ列目の「１」を抽出することを特徴とする。

【００２６】つぎの発明にかかるＬＤＰＣ符号用検査行列生成方法にあっては、基本のランダム系列のラテン方陣を作成し、ユークリッド幾何符号の第ｍ行目をｎ分割する場合、上記ラテン方陣の第ｍ行目のランダム系列をｎ分割し、当該ｎ分割後の各ランダム系列を用いてユークリッド幾何符号の第ｍ行目の「１」を抽出することを特徴とする。

【００２７】つぎの発明にかかるＬＤＰＣ符号用検査行列生成方法にあっては、基本のランダム系列のラテン方陣を複数個作成し、列方向に連結したラテン方陣群行列を用いて、ユークリッド幾何符号の第ｍ列目をｎ分割する場合、上記ラテン方陣群行列の第ｍ列目のランダム系列をｎ分割し、当該ｎ分割後の各ランダム系列を用いてユークリッド幾何符号の第ｍ列目の「１」を抽出することを特徴とする。

【００２８】つぎの発明にかかるＬＤＰＣ符号用検査行列生成方法にあっては、基本のランダム系列のラテン方陣を複数個作成し、列方向に連結したラテン方陣群行列を用いて、ユークリッド幾何符号の第ｍ行目をｎ分割する場合、上記ラテン方陣群行列の第ｍ列目のランダム系列をｎ分割し、当該ｎ分割後の各ランダム系列を用いてユークリッド幾何符号の第ｍ行目の「１」を抽出することを特徴とする。

【００２９】

【発明の実施の形態】以下に、本発明にかかるＬＤＰＣ符号用検査行列生成方法の実施の形態を図面に基づいて詳細に説明する。なお、この実施の形態によりこの発明が限定されるものではない。

【００３０】実施の形態１．本実施の形態のＬＤＰＣ符号用検査行列生成方法を説明する前に、本実施の形態のＬＤＰＣ符号用検査行列生成方法を実現可能な符号化器の位置付け、および「Ｉｒｒｅｇｕｌａｒ－ＬＤＰＣ符号」用の従来の検査行列生成方法について説明する。なお、ＬＤＰＣ符号化／復号システムの構成については、先に説明した図１３と同様である。

【００３１】送信側の符号化器１０１では、後述する本実施の形態のＬＤＰＣ符号用検査行列生成方法で検査行列Ｈを生成する。そして、以下の条件に基づいて生成行列Ｇを求める。

Ｇ：ｋ×ｎ行列（ｋ：情報長，ｎ：符号語長）

$GH^T = 0$（Ｔは転置行列）

【００３２】その後、符号化器１０１では、情報長ｋのメッセージ（$m_1 m_2 \cdots m_k$）を受け取り、上記生成行列Ｇを用いて符号語Ｃを生成する。

$C = (m_1 m_2 \cdots m_k) G$

$= (c_1 c_2 \cdots c_n)$　　（ただし、$H(c_1 c_2 \cdots c_n)^T = 0$）

【００３３】そして、変調器１０２では、生成した符号語Ｃに対して、ＢＰＳＫ，ＱＰＳＫ，多値ＱＡＭなどのデジタル変調を行い、送信する。

【００３４】一方、受信側では、復調器１０４が、通信路１０３を介して受け取った変調信号に対して、ＢＰＳＫ，ＱＰＳＫ，多値ＱＡＭなどのデジタル復調を行い、さらに、復号器１０５が、ＬＤＰＣ符号化された復調結果に対して、「ｓｕｍ－ｐｒｏｄｕｃｔアルゴリズム」による繰り返し復号を実施し、推定結果（もとの$m_1 m_2 \cdots m_k$に対応）を出力する。

【００３５】つぎに、Ｃｈｕｎｇ等（S.-Y. Chung, T. J. Richardson, and R. Urbanke, "Analysis of Sum-Product Decoding of Low-Density Parity-Check Codes Using a Gaussian Approximation," IEEE Trans. Inform.

Theory, vol.47, No.2, pp.657-670, Feb. 2001.）によって理論的に解析された、「Ｉｒｒｅｇｕｌａｒ－ＬＤＰＣ符号」用の従来の検査行列生成方法について詳細に説明する。ここでは、繰り返し復号器における入力と出力の対数尤度比（ＬＬＲ）がガウス分布に近似できると仮定してＬＤＰＣ符号の「Ｓｕｍ－Ｐｒｏｄｕｃｔアルゴリズム」を解析し、良好な行と列の重みのアンサンブルを求めている。

【００３６】なお、上記論文に記述されたＬＤＰＣ符号用検査行列生成方法であるガウス近似法（Gaussian Approximation）では、前提として、検査行列における行内の「１」の点をバリアブルノードと定義し、列内の「１」の点をチェックノードと定義する。

【００３７】まず、チェックノードからバリアブルノードへのＬＬＲメッセージ伝搬を解析する。０＜ｓ＜∞と０≦ｔ＜∞という条件において、以下の関数（２）式を定義する。なお、ｓ＝mu0はｕ０の平均値であり、ｕ０は分散値σ_n^2のガウスノイズを含む伝送路を経由して受信した信号の対数尤度比（ＬＬＲ）であり、ｔは所定の繰り返しの時点におけるチェックノードのＬＬＲ出力値のアンサンブル平均である。

【００３８】

【数２】

$$f_j(s,t)=\phi^{-1}\left(1-\left[1-\sum_{i=2}^{d_l}\lambda_i\phi(s+(i-1)t)\right]^{j-1}\right)$$

$$f(s,t)=\sum_{j=2}^{d_r}\rho_j f_j(s,t) \qquad \cdots(2)$$

$$\phi(x)=\begin{cases}1-\dfrac{1}{\sqrt{4\pi x}}\displaystyle\int_R \tanh\frac{u}{2}\cdot e^{-\frac{(u-x)^2}{4x}}du & \text{if} \quad x>0 \\ 1 & \text{if} \quad x\le 0\end{cases} \qquad \cdots(5)$$

【００４４】そして、（２）式は、等価的に下記（６）式と表すことができる。

【００４５】

【数６】

$$t_l=f(s,t_{l-1}) \qquad \cdots(6)$$

【００４６】なお、ｔｌはｌ番目の繰り返し時点におけるチェックノードのＬＬＲ出力値のアンサンブル平均である。

【００４７】ここで、誤りが０となりうるＳＮＲの限界（threshold）を求めるための条件は、ｌ→∞のときにｔｌ（ｓ）→∞（R^+と表現する）となることであり、この条件を満たすためには、以下の条件（７）式を満たす必要がある。

＊【００３９】なお、上記λ（ｘ）およびρ（ｘ）は、それぞれバリアブルノードおよびチェックノードの次数配分（バリアブルノードとチェックノードの各１行，各１列内の「１」の数を次数と表現する）の生成関数を表し、（３）式および（４）式のように表すことができる。また、λｉとρｉは、それぞれ次数ｉのバリアブルノードとチェックノードに属するエッジの比率を表す。また、ｄｌは最大バリアブルノードの次数であり、ｄｒは最大チェックノードの次数である。

【００４０】

【数３】

$$\lambda(x)=\sum_{i=2}^{d_l}\lambda_i x^{i-1} \qquad \cdots(3)$$

【００４１】

【数４】

$$\rho(x)=\sum_{i=2}^{d_r}\rho_i x^{i-1} \qquad \cdots(4)$$

【００４２】ただし、φ（ｘ）は下記（５）式のように定義する。

【００４３】

【数５】

【００４８】

【数７】

$$t<f(s,t),\ \text{全ての}\ t\in R^+ \qquad \cdots(7)$$

【００４９】つぎに、バリアブルノードからチェックノードへのＬＬＲメッセージ伝搬を解析する。０＜ｓ＜∞と０＜ｒ≦１という条件において、以下の関数（８）式を定義する。なお、ｒの初期値ｒ０はφ（ｓ）である。

【００５０】

【数８】

$$h_i(s,r)=\phi\left(s+(i-1)\sum_{j=2}^{d_r}\rho_j\phi\left(1-(1-r)^{j-1}\right)\right)$$

$$h(s,r)=\sum_{i=2}^{d_l}\lambda_i h_i(s,r) \qquad \cdots(8)$$

【００５１】そして、（８）式は、等価的に下記（９）式と表すことができる。

【００５２】

【数９】

$$r_l = h(s,r_{l-1}) \qquad \cdots(9)$$

【００５３】ここで、誤りが０となりうるＳＮＲの限界（threshold）を求めるための条件は、ｒｌ（ｓ）→０となることであり、この条件を満たすためには、以下の条件（１０）式を満たす必要がある。

【００５４】

【数１０】

$$r > h(s,r),\ \text{全ての}\ r\in(0,\phi(s)) \qquad \cdots(10)$$

【００５５】さらに、上記Ｃｈｕｎｇ等の論文では、上記式を用いて以下の手順でバリアブルノードとチェックノードの最適な次数を探索している。

（１）生成関数λ（ｘ）とガウスノイズσｎが与えられていると仮定し、生成関数ρ（ｘ）を変数として、前述した（１）式が最大となる点を探索する。なお、この探索における拘束条件は、ρ（１）＝１と正規化することと、上記（７）式を満たすことである。

（２）生成関数ρ（ｘ）とガウスノイズσｎが与えられていると仮定し（たとえば、（１）の結果より得られる値）、生成関数λ（ｘ）を変数として、（１）式が最大となる点を探索する。なお、この探索における拘束条件は、λ（１）＝１と正規化することと、上記（１０）式を満たすことである。

（３）最大「ｒａｔｅ」を求めるために、上記（１）と上記（２）を繰り返し実行し、生成関数λ（ｘ）と生成関数ρ（ｘ）のより良好なアンサンブルを線形計画法で探索する。

（４）最後に、ガウスノイズσｎより信号電力を１と正規化して、ＳＮＲの限界（threshold）を求める。

【００５６】

【数１１】

$$\text{threshold(dB)} = -10*\log10\left(2*\sigma_n^2\right) \qquad \cdots(11)$$

【００５７】しかしながら、上記Ｃｈｕｎｇ等の論文では、「ｒａｔｅ（符号化率）」の最大値により得られる検査行列が流動的になり、設計において仕様として固定されるｒａｔｅが変動してしまうため、実設計にむかない、という問題があった。また、上記ガウス近似法では、バリアブルノードの次数配分の導出とチェックノードの次数配分の導出とを所定回数にわたって繰り返し行っているため、探索処理にある程度の時間を要する、という問題もあった。

【００５８】そこで、本実施の形態においては、確定的でかつ特性が安定した「Ｉｒｒｅｇｕｌａｒ－ＬＤＰＣ符号」用の検査行列を、短時間で容易に探索する。図１は、実施の形態１のＬＤＰＣ符号用検査行列生成方法を示すフローチャートである。

（１）「ｒａｔｅ」が与えられているものと仮定する。すなわち、要求「ｒａｔｅ」を固定する。実際の設計では、目標「ｒａｔｅ」が予め指定されている場合が多いためである。

（２）生成関数λ（ｘ）と生成関数ρ（ｘ）を同時に変数として扱い、ガウスノイズσｎが最大になるように、線形計画法で最適な生成関数λ（ｘ）と生成関数ρ（ｘ）を探索する。この探索における拘束条件は、λ（１）＝１，ρ（１）＝１と正規化し、さらに上記（１０）式を満たすことである。

【００５９】このように、本実施の形態では、上記（９）式と上記（１０）式を満たす生成関数λ（ｘ）と生成関数ρ（ｘ）を１回の線形計画法で求めることとしたため、上記論文のように、生成関数λ（ｘ）の導出と生成関数ρ（ｘ）の導出を繰り返し実行し、双方の最適値を求める方法よりも、容易かつ短時間に、確定的でかつ特性が安定したＬＤＰＣ符号用の検査行列を生成することができる。

【００６０】実施の形態２．実施の形態２では、前述の実施の形態１においてユークリッド幾何符号を採用し、１行あるいは１列の「１」の配置を分割することによって、「Ｉｒｒｅｇｕｌａｒ－ＬＤＰＣ符号」の検査行列を生成する。

【００６１】まず、実施の形態１におけるＬＤＰＣ符号用検査行列生成方法により、生成関数λ（ｘ）と生成関数ρ（ｘ）のアンサンブルを導出する。図２は、ｒａｔｅ＝０．５とした場合の生成関数λ（ｘ）と生成関数ρ（ｘ）のアンサンブルを示す図である。なお、σＧＡはガウス近似法により導出した「threshold」時のノイズ分散値を表し、ＳＮＲｎｏｒｍ（ＧＡ）はガウス近似法により導出した「threshold」のＳＮＲとシャノン限界のＳＮＲとの差分を表し、ｘは重みを表し、λｘおよびρｘはそれぞれバリアブルノードとチェックノードの重み配分を表す。

【００６２】また、基準になるユークリッド幾何符号は、ＥＧ（２，2^5）を想定し、ｄｌ＝３２とした。また、重み配分λｘのｘの値および重み配分ρｘのｘの値は、組み合わせで３２（ｄｌ）を構成できる値とする。

【００６３】図３は、ＥＧ（２，2^5）を想定し、ｄｌ＝３２とした場合の、分割テーブルを示す図である。図３に示すとおり、図２のｘは、組み合わせにより必ず３２となる。たとえば、図示の７ｘ４と２ｘ２の組み合わせ

は、重みが３２の１列を、重みが７の４列と重みが２の２列に分割可能なことを表している。このように、ＥＧ（2，2^5）の符号を基本として、図３のように重みが３２の各行列を適切に分割すると、「Ｉｒｒｅｇｕｌａｒ－ＬＤＰＣ符号」の検査行列を構成することができる。

【００６４】ここで、分割処理を行う前に、図２に示す生成関数λ（ｘ）と生成関数ρ（ｘ）のアンサンブルの重み配分を以下の手順で調整する。図４は、重み配分調整用テーブルを示す図である。なお、ＥＧ（2，2^5）のユークリッド幾何符号は、１０２３行×１０２３列で構成される。

【００６５】（１）ガウス近似法で求めた生成関数λ（ｘ）と生成関数ρ（ｘ）のアンサンブル（表１参照）をテーブルの２列目と３列目に設定する。

（２）重み配分λ_xおよびρ_x（３列目）と、ＥＧ（2，2^5）における全行列の「１」の総数ＴＰ＝３２７３６と、を乗算し、重み単位の重み総数を求め、さらに、当該重み単位の重み総数とその総和を４列目に設定する。

（３）重み単位の重み総数（４列目）を対応する重みｘで割り、重み単位の総列数を求め、それを５列目に設定する。

（４）重み単位の総列数が小数点以下を含む場合、丸め処理（四捨五入，切上げ，切捨て等）を行い、その結果を６列目に設定する。

（５）丸め処理後の重み単位の総列数（６列目）と対応する重みｘとを乗算し、丸め処理後の重み単位の重み総数を求め、それを７列目に設定する。そして、各重み総数の総和（７列目の合計の行）が行列内の「１」の総数（ＴＰ＝３２７３６）と等しいかどうかを確認する。

（６）行列内の「１」の総数に等しくない場合、丸め処理後の重み単位の重み総数（７列目）を整数単位で調整し、その結果を８列目に設定する。この場合、８列目の総和が、行列内の「１」の総数（ＴＰ＝３２７３６）に等しくなるように調整する。

（７）調整後の重み単位の重み総数（８列目）を対応する重みｘで割り、調整後の重み単位の総列数を求め、それを９列目に設定する。調整後の各重みの配分（１１列目）は、可能な限りガウス近似法で求めた値（３列目）に近い値にする。

【００６６】図５は、重み配分後の生成関数λ（ｘ）と生成関数ρ（ｘ）のアンサンブルを示す図である。

【００６７】つぎに、ユークリッド幾何符号における１行あるいは１列の分割手順について説明する。

【００６８】たとえば、分割手順に関して、Ｙ．Ｋｏｕ等の論文では、規則的に分割する方法を提示している。図６は、上記論文における分割手順を示す図である。まず、図６に示すように行列のナンバリングを行う。ここでは、列番号を左端から順に１，２，３，…とし、行番号を上から順に１，２，３，…とする。そして、たとえば、３２点×１列を８点×４列に分割する場合、下記（１２）式にしたがって規則的に分割する。

【００６９】

【数１２】

$$S_m(n)=B_l(m+4*n) \quad \cdots(12)$$

【００７０】なお、ｍ＝１，２，３，４とし、ｎ＝０，１，２，３，４，５，６，７とし、ｌはＥＧ（2，2^5）の列番号を表す。また、B_l（ｘ）はＥＧ（2，2^5）のｌ列目の「１」の位置を表し、S_m（ｎ）は分割後の行列のｍ列目の「１」の位置を表す。

【００７１】具体的にいうと、ＥＧ（2，2^5）における１列中の「１」の位置を示す行番号は、

B_l（ｘ）＝{1 32 114 136 149 223 260 382 402 438 467 507 574 579 588 622 634 637 638 676 717 728 790 851 861 879 947 954 971 977 979 998}

となり、その結果、分割後の行列における１～４列目の「１」の位置を示す行番号は、B_l（ｘ）から「１」の番号が規則的に抽出され、

S_1（ｎ）＝{1 149 402 574 634 717 861 971}

S_2（ｎ）＝{32 223 438 579 637 728 879 977}

S_3（ｎ）＝{114 260 467 588 638 790 947 979}

S_4（ｎ）＝{136 382 507 622 676 851 954 998}

となる。すなわち、３２点×１列が８点×４列に分割される。

【００７２】一方、本実施の形態におけるユークリッド幾何符号の分割処理は、上記のように規則的に分割するのではなく、B_l（ｘ）から「１」の番号をランダムに抽出する。なお、この抽出処理は、ランダム性が保持されるのであればどのような方法を用いてもよい。

【００７３】したがって、分割後の行列のｍ列目の「１」の位置の一例をR_m（ｎ）とした場合、R_m（ｎ）は、

R_1（ｎ）＝{1 114 574 637 851 879 977 979}

R_2（ｎ）＝{32 136 402 467 588 728 861 971}

R_3（ｎ）＝{149 260 382 438 579 638 717 998}

R_4（ｎ）＝{223 507 622 634 676 790 947 954}

となる。

【００７４】上記のような本実施の形態の分割手順をグラフ上で表現すると、以下のように表現することができる。図７は、分割前のＥＧ（2，2^5）のグラフを示す図である。なお、両ノードを結ぶ線はエッジと表現する。図７では、分割前の１０２３行×１０２３列（各行列の重みがそれぞれ３２）のユークリット幾何符号を表現している。また、図８は、ＥＧ（2，2^5）のエッジをランダムに選択した、分割後のグラフを示す図である。

【００７５】ここで、上記で説明したＬＤＰＣ符号の特性を比較する。図９は、Ｅｂ／Ｎｏ（情報１ビットあたりの信号電力対ノイズ電力比）と誤り率特性（ＢＥＲ）

との関係を示す図である。なお、繰り返し回数は５０回で、復号法は「Ｓｕｍ－Ｐｒｏｄｕｃｔアルゴリズム」である。

【００７６】なお、図中"Simple regular extended EG(2,2^5)"は、Ｙ．Ｋｏｕ等の発案によるＥＧ（２，2^5）の規則的な列の分割を実施した場合の、ｒａｔｅ＝０．５の「Ｒｅｇｕｌａｒ－ＬＤＰＣ符号」であり、"Random regular extended EG(2,2^5)"は、本実施の形態によるＥＧ（２，2^5）のランダムな列の分割を実施した場合の、ｒａｔｅ＝０．５の「Ｒｅｇｕｌａｒ－ＬＤＰＣ符号」である。図１０は、これらの「Ｒｅｇｕｌａｒ－ＬＤＰＣ符号」のアンサンブルを示す図である。

【００７７】また、図中"Simple irregular extended EG(2,2^5)"は、実施の形態１の方法によって特定されたアンサンブルに対して、Ｙ．Ｋｏｕ等の発案によるＥＧ（２，2^5）の規則的な列の分割を実施した場合の、ｒａｔｅ＝０．５の「Ｉｒｒｅｇｕｌａｒ－ＬＤＰＣ符号」であり、"Random irregular extended EG(2,2^5)"は、実施の形態１の方法によって特定されたアンサンブルに対して、本実施の形態によるＥＧ（２，2^5）のランダムな列の分割を実施した場合の、ｒａｔｅ＝０．５の「Ｉｒｒｅｇｕｌａｒ－ＬＤＰＣ符号」である。図１１は、これらの「Ｉｒｒｅｇｕｌａｒ－ＬＤＰＣ符号」のアンサンブルを示す図である。

＊【００７８】図９からわかるように、同一レートでは、「Ｒｅｇｕｌａｒ－ＬＤＰＣ符号」より「Ｉｒｒｅｇｕｌａｒ－ＬＤＰＣ符号」のほうが性能がよい。また、Ｙ．Ｋｏｕ等の論文のような規則的な分割では、「Ｉｒｒｅｇｕｌａｒ－ＬＤＰＣ符号」であっても大幅な改善は見込めないが、本実施の形態のランダムな分割を実施すると性能が画期的に改善される。

【００７９】つぎに、上記ランダム分割の一例を詳細に説明する。ここでは、ランダム分割を行う場合のランダム系列を容易かつ確定的に生成する。この方法による利点は、送信側と受信側が同じランダム系列を生成できることにある。これは、現実のシステムではきわめて重要となる。また、符号特性の条件が正確に規定できる、という利点もある。

【００８０】（１）基本のランダム系列を作成する。以下に、ランダム系列作成の一例を記述する。ここでは、説明の便宜上、ユークリッド幾何符号ＥＧ（２，2^4）を用いる。ユークリッド幾何符号ＥＧ（２，2^4）の場合、１行に存在する「１」の数は2^4個である。

【００８１】ＰをＰ≧2^sを満たす最小の素数とした場合、たとえば、2^4のときはＰ＝１７となる。ここで、系列長Ｐ－１＝１６の基本のランダム系列Ｃ（ｉ）を（１３）式にしたがって作成する。

$$C(1)=1$$
$$C(i+1)=G_0\times C(i) \mod P \quad \cdots(13)$$

ただし、ｉ＝１，…，Ｐ－１とし、G_0はガロア体ＧＦ（Ｐ）の原始元である。その結果、Ｃ（ｉ）は、

Ｃ（ｉ）＝{1 3 9 10 13 5 15 11 16 14 8 7 4 12 2 6}

となる。なお、2^4の場合は、Ｐが2^4＋１となり、ランダム系列長が１６になるので問題ないが、2^5の場合等は、Ｐが2^5＋１以上になりランダム系列長が2^5を超えてしまう。そのような場合は、2^5を超える数値をランダム系列から削除することで対応する。

【００８２】そして、ランダム系列Ｃ（ｉ）を巡回シフトして、ランダム系列のラテン方陣を生成する。図１２は、ランダム系列のラテン方陣を示す図である。

【００８３】（２）ランダム系列のラテン方陣を用いて、ユークリッド幾何符号ＥＧ（２，2^4）の１列目の「１」の位置を示す行番号の配列B_1（ｘ）から、「１」の番号をランダムに抽出し、ユークリッド幾何符号の第１列目を分割する。なお、上記配列B_1（ｘ）は、

B_1（ｘ）＝{1 14 16 19 45 49 55 107 115 126 127 182 210 224 231 247}

である。

【００８４】たとえば、ユークリッド幾何符号の第１列目を４分割する場合（各列の重みが４となる）、上記ランダム系列のラテン方陣の１行目から、４つのランダム系列L_1（ｎ）～L_4（ｎ）を抽出する。

L_1（ｎ）＝{1,3,9,10}
L_2（ｎ）＝{13,5,15,11}
L_3（ｎ）＝{16,14,8,7}
L_4（ｎ）＝{4,12,2,6}

【００８５】そして、４つのランダム系列L_1（ｎ）～L_4（ｎ）を用いて、配列B_1（ｘ）から、「１」の番号をランダムに抽出する。その結果、Ｒ（L_1（ｎ））～Ｒ（L_4（ｎ））は、

Ｒ（L_1（ｎ））＝{1,16,115,126}
Ｒ（L_2（ｎ））＝{210,45,231,127}
Ｒ（L_3（ｎ））＝{247,224,107,55}
Ｒ（L_4（ｎ））＝{19,182,14,49}

となる。

【００８６】（３）同様に、ランダム系列のラテン方陣を用いて、ユークリッド幾何符号ＥＧ（２，2^4）の２列目の「１」の位置を示す行番号の配列B_2（ｘ）から、「１」の番号をランダムに抽出し、ユークリッド幾何符号の第２列目を分割する。なお、上記配列B_2（ｘ）は、

B_2（ｘ）＝{2 15 17 20 46 50 56 108 116 127 128 183 211 225 232 248}

である。

【００８７】たとえば、ユークリッド幾何符号の第２列

目を４分割する場合（各列の重みが４となる）、上記ランダム系列のラテン方陣の２行目から、４つのランダム系列L_1（ｎ）～L_4（ｎ）を抽出する。

L_5（ｎ）＝｛3, 9, 10, 13｝

L_6（ｎ）＝｛5, 15, 11, 16｝

L_7（ｎ）＝｛14, 8, 7, 4｝

L_8（ｎ）＝｛12, 2, 6, 1｝

【００８８】そして、４つのランダム系列L_5（ｎ）～L_8（ｎ）を用いて、配列B_2（ｘ）から、「１」の番号をランダムに抽出する。その結果、Ｒ（L_5（ｎ））～Ｒ（L_8（ｎ））は、

Ｒ（L_5（ｎ））＝｛17, 116, 127, 211｝

Ｒ（L_6（ｎ））＝｛46, 232, 128, 248｝

Ｒ（L_7（ｎ））＝｛225, 108, 56, 20｝

Ｒ（L_8（ｎ））＝｛183, 15, 50, 2｝

となる。以降、同様の手順でユークリッド幾何符号の全ての列を分割する。

【００８９】（４）なお、上記ランダム系列のラテン方陣でランダム系列が不足するような場合は、（１３）式の基本のランダム系列のG_0に原始元より大きくかつ2^s＊

$$C(1)=0$$

$$C(i+1)=G_0\times C(i) \mod P \quad \cdots(14)$$

ただし、ｉ＝１，…，Ｐ－１とし、G_0はガロア体ＧＦ（Ｐ）の原始元である。その結果、Ｃ（ｉ）は、

Ｃ（ｉ）＝｛0, 1, 3, 9, 27, 19, 26, 17, 20, 29, 25, 13, 8, 24, 10, 30, 28, 22, 4, 12, 5, 15, 14, 11, 2, 6, 18, 23, 7, 21｝

となる。

【００９４】つぎに以下の操作をする。

Ｃ（ｉ）＝Ｃ（ｉ）＋１，　Ｃ（３２）＝３２

その結果、Ｃ（ｉ）は、

Ｃ（ｉ）＝｛1, 2, 4, 10, 28, 20, 27, 17, 18, 21, 30, 26, 14, 9, 25, 11, 31, 29, 23, 5, 13, 6, 16, 15, 12, 3, 7, 19, 24, 8, 22, 32｝

となる。これを図１７の左側の太枠内に示し、基本のランダム系列とする。

【００９５】つぎに、Ｃ（ｉ）のランダム系列をある間隔Ｓ（ｊ），ｊ＝１，２，…，Ｐ－１で読む方法について説明する。この間隔は、Ｐ－１個生成可能である。ここではＰ－１＝３０である。

Ｓ（ｊ）＝｛1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30｝

【００９６】ランダム系列を一定の間隔で飛ばし読みする系列をＬＢｊ（ｉ）とすると、

ＬＢｊ（ｉ）＝（（Ｓ（ｊ）＊ｉ）　ｍｏｄ　Ｐ）＋１

ただし、ｊ＝１，２，…，Ｐ－１であり、ｉ＝１，…，Ｐ－１である。

【００９７】たとえば　ｊ＝１の場合、

LB_1（１）＝｛2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31｝

となる。

【００９８】ここで、ランダム系列の系列数３２に対

＊以下の素数を代入して再度基本のランダム系列を作り、同様の手順で分割する。

【００９０】つぎに、もう一つのランダム系列のラテン方陣の作成法について説明する。ここでは、説明の便宜上、ユークリッド幾何符号ＥＧ（２，2^5）を用いる。ユークリッド幾何符号ＥＧ（２，2^5）の場合、１行に存在する「１」の数は2^5個である。

【００９１】上記例では、ＥＧ（２，2^4）のため１６（行数）×１６（列数）のラテン方陣を作成したが、これでは不足する場合がある。ここでは、ＥＧ（２，2^5）を例にとり、３２（行数）×９６０（列数）まで拡張する。行列のサイズは、３２（行数）×３２（列数）のラテン方陣の組を３０組列方向に並べたことにより決められる。

【００９２】また、上記の例では、ＰをＰ≧2^sを満たす最小の素数としたが、ここでは、ＰをＰ≧2^sを満たす最大の素数とする。たとえば、2^5のときはＰ＝３１となる。

【００９３】ここで、系列長Ｐ＝３１の基本のランダム系列Ｃ（ｉ）を（１４）式に従って作成する。

し、１とＰから2^sまでの整数が不足している。この場合は１，３２が不足している。

$LB_j(j)=32$，$LB_j(32-j)=1$

を挿入する。

【００９９】この結果、

LB_1（ｉ）＝｛32, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 1, 31｝

となる。

【０１００】同様にｊ＝２の場合、

LB_2（ｉ）＝｛3, 32, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 1, 28, 30｝

【０１０１】ここからラテン方陣を作成する。L_{jq}（ｉ）はｊ番目のラテン方陣のｑ列目の系列である。

$L_{jq}(i)=LB_j(i)$とすると、たとえば、

$L_{11}(i)=C(LB_1(i))$＝｛32, 2, 4, 10, 28, 20, 27, 17, 18, 21, 30, 26, 14, 9, 25, 11, 31, 29, 23, 5, 13, 6, 16, 15, 12, 3, 7, 19, 24, 8, 1, 22｝

となる。これは、図１８の１列目がL_{11}（ｉ）に相当する。

【０１０２】このL_{11}（ｉ）を巡回シフトしてL_{1q}（ｉ）の３２（行数）×３２（列数）のラテン方陣を作成する。同様の手順で、ラテン方陣をL_{2q}（ｉ），L_{3q}（ｉ），…，L_{30q}（ｉ）まで作成し、３２（行数）×（３２×３０）（列数）のラテン方陣の組み合わせを作成する。

【０１０３】ここで、具体例を説明する。ユークリッド

17

幾何符号ＥＧ（２，2^5）の１列目の「１」の位置を示す行番号の配列Ｂ１（ｘ）から「１」の番号をランダムに抽出し、ユークリッド幾何符号の第１列目を分解する。なお、上記配列Ｂ１（ｘ）は、

Ｂ１（ｘ）＝{1 32 114 136 149 223 260 382 402 438 467 507 574 579 588622 634 637 638 676 717 728 790 851 861 879 947 954 971 977 979 998}

である。

【０１０４】たとえば、ユークリッド幾何符号の第１列目を４分割する場合（各列の重みが８となる）、上記ランダム系列のラテン方の陣組み合わせＬ$_{jq}$（ｉ）の１番＊目のラテン方陣の１列目から４つのランダム系列Ｌ１（ｎ）～Ｌ４（ｎ）を抽出する。

Ｌ１（ｎ）＝{32, 2, 4, 10, 28, 20, 27, 17}

Ｌ２（ｎ）＝{18, 21, 30, 26, 14, 9, 25, 11}

Ｌ３（ｎ）＝{31, 29, 23, 5, 13, 6, 16, 15, }

Ｌ４（ｎ）＝{12, 3, 7, 19, 24, 8, 1, 22}

【０１０５】そして、４つのランダム系列Ｌ１（ｎ）～Ｌ４（ｎ）を用いて、配列Ｂ１（ｘ）から「１」の番号をランダムに抽出する。その結果Ｒ（Ｌ１（ｎ））～Ｒ（Ｌ４（ｎ））は、

Ｒ（Ｌ１（ｎ））＝{998, 32, 136, 438, 954, 676, 947, 634}

Ｒ（Ｌ２（ｎ））＝{637, 717, 977, 879, 579, 402, 861, 467}

Ｒ（Ｌ３（ｎ））＝{979, 971, 790, 149, 574, 223, 622, 588}

Ｒ（Ｌ４（ｎ））＝{507, 114, 260, 638, 851, 382, 1, 728}

となる。以降、同様の手順でユークリッド幾何符号の全ての列を分割する。

【０１０６】

【発明の効果】以上、説明したとおり、本発明によれば、生成関数λ（ｘ）と生成関数ρ（ｘ）を１回の線形計画法で求めることとしたため、上記論文のように、生成関数λ（ｘ）の導出と生成関数ρ（ｘ）の導出を繰り返し実行し、双方の最適値を求める方法よりも、容易かつ短時間に、確定的でかつ特性が安定したＬＤＰＣ符号用の検査行列を生成することができる、という効果を奏する。

【０１０７】つぎの発明によれば、同一レートでは「Ｒｅｇｕｌａｒ－ＬＤＰＣ符号」よりも良好な特性を得ることができる、という効果を奏する。また、規則的な分割では、「Ｉｒｒｅｇｕｌａｒ－ＬＤＰＣ符号」であっても大幅に特性を改善することはできないが、ランダムな分割を実施すると、特性を画期的に改善することができる、という効果を奏する。

【０１０８】つぎの発明によれば、重み配分を、重み単位の重み総数が整数で、かつ重み単位の重み総数の総和とユークリット幾何符号の「１」の総数とが等しくなるように調整することによって、より高精度な分割処理を実現できる、という効果を奏する。

【０１０９】つぎの発明によれば、送信側と受信側が同じランダム系列を生成することができる、という効果を奏する。また、ランダム系列のラテン方陣を作成することによって、符号特性の条件を正確に規定できる、という効果を奏する。

【０１１０】つぎの発明によれば、送信側と受信側が同じランダム系列を生成することができる、という効果を奏する。また、ランダム系列のラテン方陣を作成することによって、符号特性の条件を正確に規定できる、という効果を奏する。

【０１１１】つぎの発明によれば、送信側と受信側が同じランダム系列を生成することができる、という効果を奏する。また、ランダム系列のラテン方陣を複数個作成することによって、符号特性の条件を正確に規定できる、という効果を奏する。

【０１１２】つぎの発明によれば、送信側と受信側が同じランダム系列を生成することができる、という効果を奏する。また、ランダム系列のラテン方陣を複数個作成することによって、符号特性の条件を正確に規定できる、という効果を奏する。

【図面の簡単な説明】

【図１】　実施の形態１のＬＤＰＣ符号用検査行列生成方法を示すフローチャートである。

【図２】　ｒａｔｅ＝０．５とした場合の生成関数λ（ｘ）と生成関数ρ（ｘ）のアンサンブルを示す図である。

【図３】　ＥＧ（２，2^5）を想定し、ｄ１＝３２とした場合の、分割テーブルを示す図である。

【図４】　重み配分調整用テーブルを示す図である。

【図５】　重み配分後の生成関数λ（ｘ）と生成関数ρ（ｘ）のアンサンブルを示す図である。

【図６】　従来の分割手順を示す図である。

【図７】　分割前のＥＧ（２，2^5）のグラフを示す図である。

【図８】　分割後のＥＧ（２，2^5）のグラフを示す図である。

【図９】　Ｅｂ／Ｎｏと誤り率特性との関係を示す図である。

【図１０】　「Ｒｅｇｕｌａｒ－ＬＤＰＣ符号」のアンサンブルを示す図である。

【図１１】　「Ｉｒｒｅｇｕｌａｒ－ＬＤＰＣ符号」のアンサンブルを示す図である。

【図１２】　ランダム系列のラテン方陣を示す図である。

【図１３】　ＬＤＰＣ符号化／復号システムを示す図である。

【図１４】　従来のＬＤＰＣ符号用の検査行列を示す図

である。

【図１５】　ユークリッド幾何符号ＥＧ（２，2^2）の構成を示す図である。

【図１６】　列の重みを４から２に規則的に分離した例を示す図である。

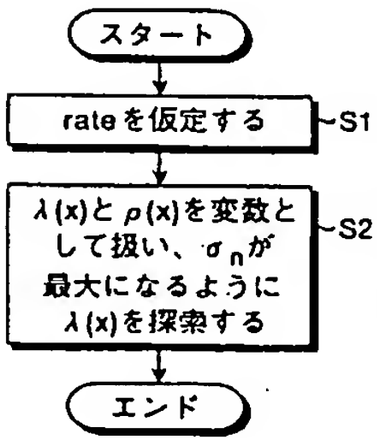
【図１７】　ランダム系列のラテン方陣を示す図である。

【図１８】　ランダム系列のラテン方陣を示す図である。

【符号の説明】

１０１　符号化器、１０２　変調器、１０３　通信路、１０４　復調器、１０５　復号器。

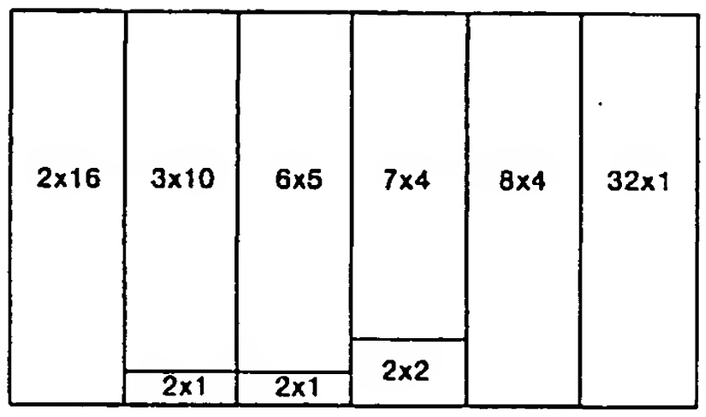
【図１】



【図２】

d_l	32	
rate	0.5	
	x	λ_x
	2	0.178783
	3	0.149148
	6	0.019896
	7	0.220144
	8	0.005388
	32	0.426641
	x	ρ_x
	10	0.333333
	11	0.666667
σ_{GA}	0.962089	
SNR_{norm}(GA)	0.1486 dB	

【図３】



【図４】

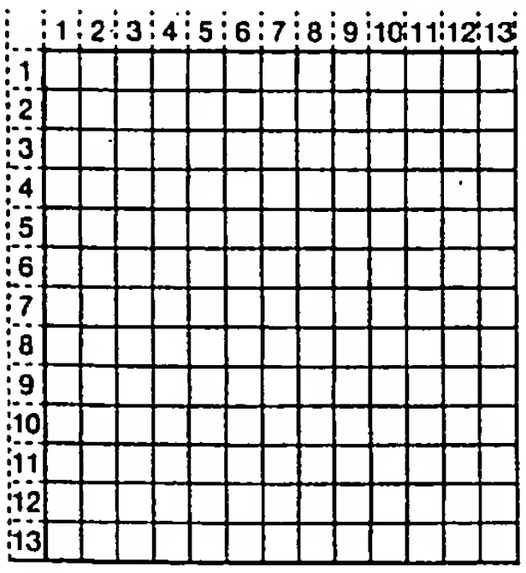
1	2	3	4	5	6	7	8	9	10	11
	x	λx							x	λx
チェック	2	0.178783	5852.64	2926.32	2926	5852	5846	2923	2	0.178580156
ノード	3	0.149148	4882.509	1627.503	1628	4884	4890	1630	3	0.149376833
(列)	6	0.019896	651.3155	108.5526	109	654	660	110	6	0.02016129
	7	0.220144	7206.634	1029.519	1030	7210	7196	1028	7	0.219819159
	8	0.005388	176.3816	22.0477	22	176	192	24	8	0.005865103
	32	0.426641	13966.52	436.4537	436	13952	13952	436	32	0.426197458
合計		1	32736	6150.396	6151	32728	32736	6151		1
	x	ρx							x	ρx
バリアブル	10	0.3125	10230	1023	1023	10230	10230	1023	10	0.3125
ノード(行)	11	0.6875	22506	2046	2046	22506	22506	2046	11	0.6875
合計		1	32736	3069	3069	32736	32736	3069		0.3125
rate		0.501007762								0.501056739

・行列内の「1」の総数TP=1023×32=32736

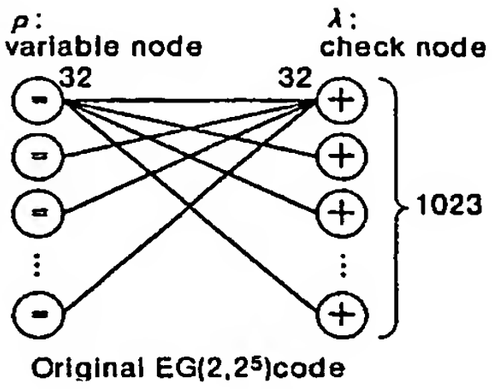
【図５】

d_l	32		
rate	0.5		
	x	λ_x	No.
	2	0.178580156	2923
	3	0.149376833	1630
	6	0.02016129	110
	7	0.219819159	1028
	8	0.005865103	24
	32	0.426197458	436
	x	ρ_x	No.
	10	0.333333	1023
	11	0.666667	2046
σ_{GA}	0.962089		
SNR_{norm}(GA)	0.1486 dB		

【図６】



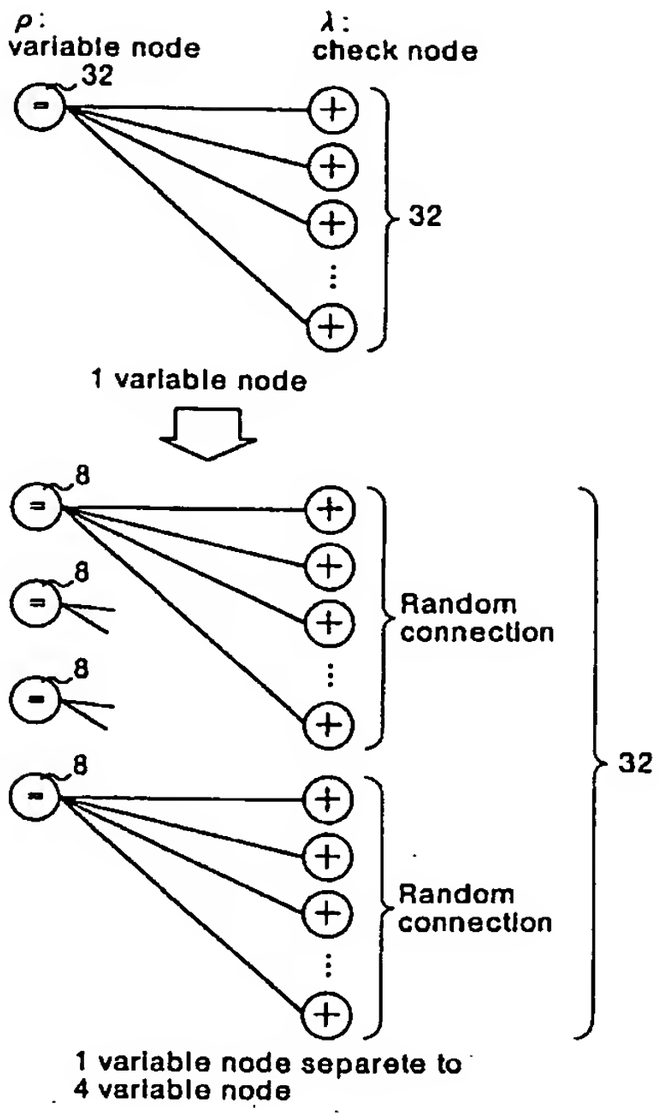
【図７】



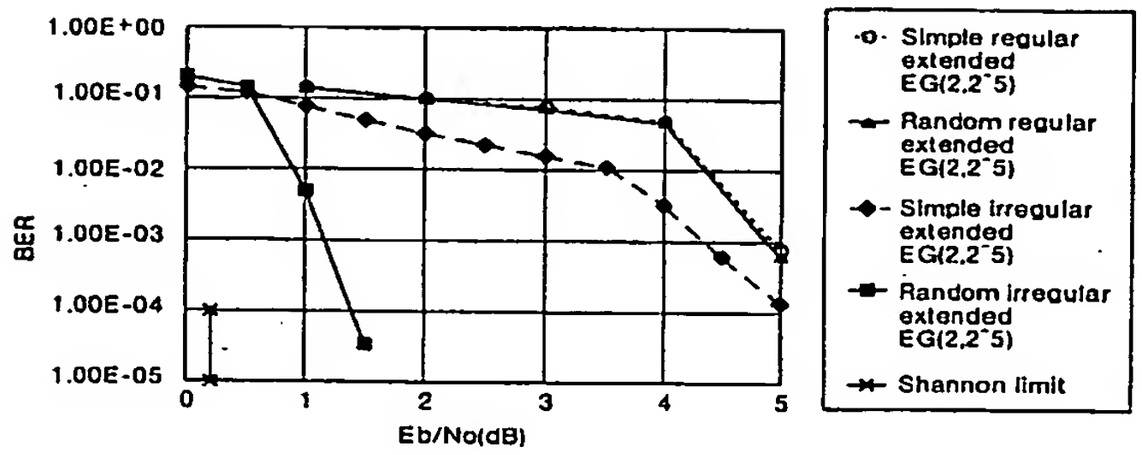
【図１０】

d_l	32		
rate	0.5		
	x	λ_x	No.
	16	1	2046
	x	ρ_x	No.
	32	1	1023

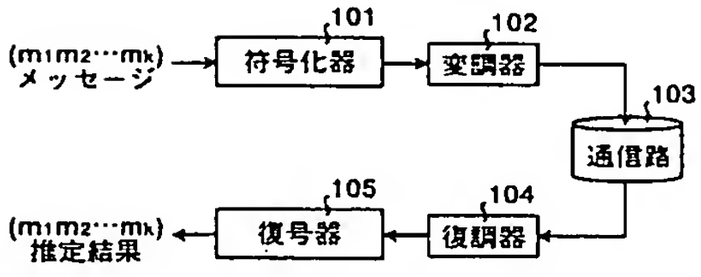
【図８】



【図９】



【図１３】



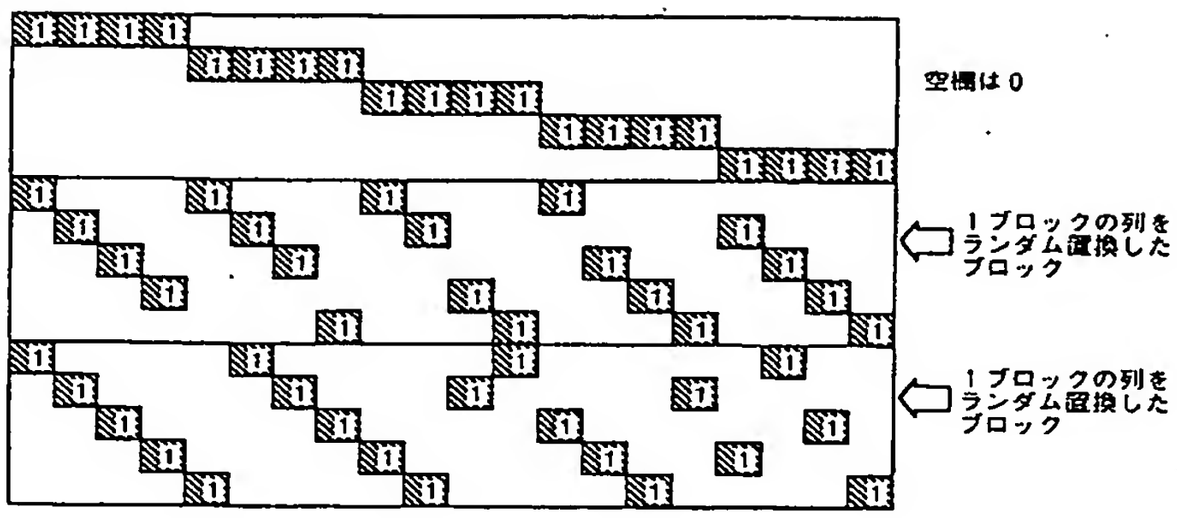
【図１１】

d_c	32		
rate	0.5		
	x	λ_x	No.
	2	0.178580156	2923
	3	0.149376833	1630
	6	0.02016129	110
	7	0.219819159	1028
	8	0.005865103	24
	32	0.426197458	436
	x	ρ_x	No.
	10	0.3125	1023
	11	0.6875	2046

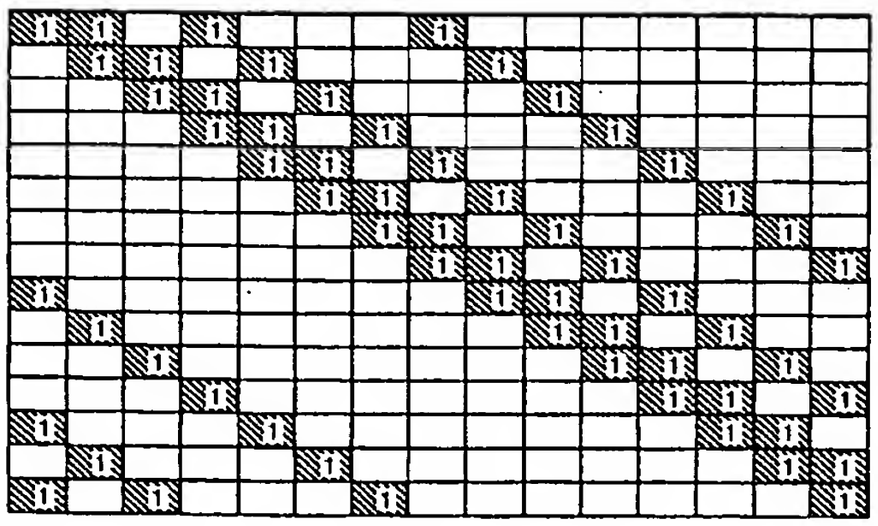
【図１２】

1	3	9	10	13	5	15	11	16	14	8	7	4	12	2	6
3	9	10	13	5	15	11	16	14	8	7	4	12	2	6	1
9	10	13	5	15	11	16	14	8	7	4	12	2	6	1	3
10	13	5	15	11	16	14	8	7	4	12	2	6	1	3	9
13	5	15	11	16	14	8	7	4	12	2	6	1	3	9	10
5	15	11	16	14	8	7	4	12	2	6	1	3	9	10	13
15	11	16	14	8	7	4	12	2	6	1	3	9	10	13	5
11	16	14	8	7	4	12	2	6	1	3	9	10	13	5	15
16	14	8	7	4	12	2	6	1	3	9	10	13	5	15	11
14	8	7	4	12	2	6	1	3	9	10	13	5	15	11	16
8	7	4	12	2	6	1	3	9	10	13	5	15	11	16	14
7	4	12	2	6	1	3	9	10	13	5	15	11	16	14	8
4	12	2	6	1	3	9	10	13	5	15	11	16	14	8	7
12	2	6	1	3	9	10	13	5	15	11	16	14	8	7	4
2	6	1	3	9	10	13	5	15	11	16	14	8	7	4	12
6	1	3	9	10	13	5	15	11	16	14	8	7	4	12	2

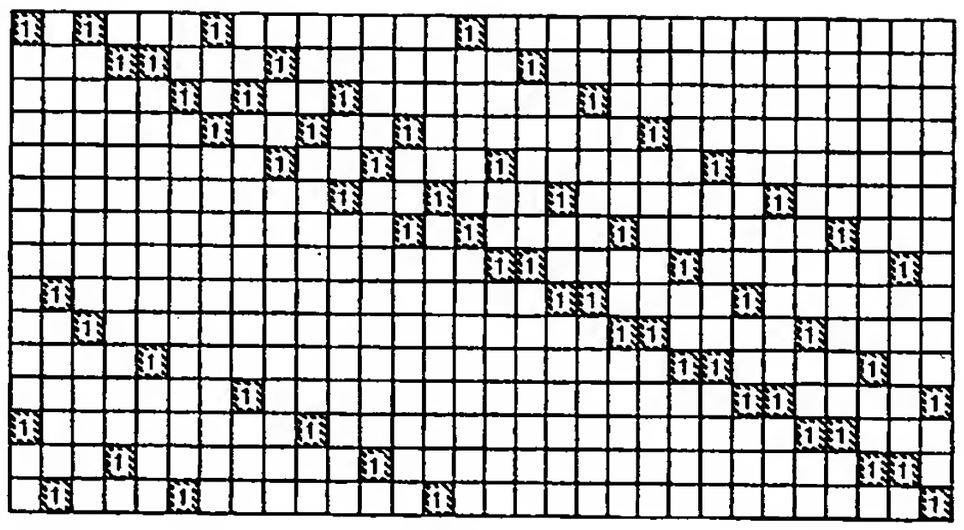
【図１４】



【図１５】



【図１６】



【図１７】

GO 3

		j.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
C(1)	1	LB(1)	32	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
C(2)	2	LB(2)	2	32	7	9	11	13	15	17	19	21	23	25	27	29	31	2	4	6	8	10	12	14	16	18	20	22	24	26	28	1
C(3)	4	LB(3)	3	5	32	13	16	19	22	25	28	31	3	6	9	12	15	18	21	24	27	30	2	5	8	11	14	17	20	23	1	30
C(4)	10	LB(4)	4	7	10	32	21	25	29	2	6	10	14	18	22	26	30	3	7	11	15	19	23	27	31	4	8	12	16	1	26	29
C(5)	28	LB(5)	5	9	13	17	32	31	5	10	15	20	25	30	4	9	14	19	24	29	3	8	13	18	23	28	2	7	1	20	24	28
C(6)	20	LB(6)	6	11	16	21	26	32	12	18	24	30	5	11	17	23	29	4	10	16	22	28	3	9	15	21	27	1	12	17	22	27
C(7)	27	LB(7)	7	13	19	25	31	6	32	26	2	9	16	23	30	6	13	20	27	3	10	17	24	31	7	14	1	2	8	14	20	26
C(8)	17	LB(8)	8	15	22	29	5	12	19	32	11	19	27	4	12	20	28	5	13	21	29	6	14	22	30	1	21	28	4	11	18	25
C(9)	16	LB(9)	9	17	25	2	10	18	26	3	32	29	7	16	25	3	12	21	30	8	17	26	4	13	1	7	15	23	31	8	16	24
C(10)	21	LB(10)	10	19	28	6	15	24	2	11	20	32	18	28	7	17	27	6	16	26	5	15	25	1	22	31	9	18	27	5	14	23
C(11)	30	LB(11)	11	21	31	10	20	30	9	19	29	8	32	9	20	31	11	22	2	13	24	4	1	4	14	24	3	13	23	2	12	22
C(12)	26	LB(12)	12	23	3	14	25	5	16	27	7	18	29	32	2	14	26	7	19	31	12	1	15	26	6	17	28	8	19	30	10	21
C(13)	14	LB(13)	13	25	6	18	30	11	23	4	16	28	9	21	32	28	10	23	5	18	1	24	5	17	29	10	22	3	15	27	8	20
C(14)	9	LB(14)	14	27	9	22	4	17	30	12	25	7	20	2	15	32	25	8	22	1	31	13	26	8	21	3	16	29	11	24	6	19
C(15)	25	LB(15)	15	29	12	26	9	23	6	20	3	17	31	14	28	11	32	24	1	5	19	2	16	30	13	27	10	24	7	21	4	18
C(16)	11	LB(16)	16	31	15	30	14	29	13	28	12	27	11	26	10	25	9	1	8	23	7	22	6	21	5	20	4	19	3	18	2	17
C(17)	31	LB(17)	17	2	18	3	19	4	20	5	21	6	22	7	23	8	1	32	25	10	26	11	27	12	28	13	29	14	30	15	31	16
C(18)	29	LB(18)	18	4	21	7	24	10	27	13	30	16	2	19	5	1	24	9	32	28	14	31	17	3	20	6	23	9	26	12	29	15
C(19)	23	LB(19)	19	6	24	11	29	16	3	21	8	26	13	31	1	22	8	25	11	32	2	20	7	25	12	30	17	4	22	9	27	14
C(20)	5	LB(20)	20	8	27	15	3	22	10	29	17	5	24	1	18	5	23	10	28	15	32	9	26	16	4	23	11	30	18	6	25	13
C(21)	13	LB(21)	21	10	30	18	8	28	17	6	26	15	1	12	31	19	7	26	14	2	21	32	18	7	27	16	5	25	14	3	23	12
C(22)	6	LB(22)	22	12	2	23	13	3	24	14	4	1	4	24	13	2	22	11	31	20	9	29	32	29	19	9	30	20	10	31	21	11
C(23)	16	LB(23)	23	14	5	27	18	8	31	22	1	25	15	5	26	16	6	27	17	7	28	18	8	32	11	2	24	15	6	28	19	10
C(24)	15	LB(24)	24	16	8	31	23	15	7	1	13	4	26	17	8	30	21	12	3	25	16	7	29	20	32	26	18	10	2	25	17	9
C(25)	12	LB(25)	25	18	11	4	28	21	1	30	22	14	6	29	21	13	5	28	20	12	4	27	19	11	3	32	12	5	29	22	15	8
C(26)	3	LB(26)	26	20	14	8	2	1	14	7	31	24	17	10	3	27	20	13	6	30	23	16	9	2	26	19	32	31	25	19	13	7
C(27)	7	LB(27)	27	22	17	12	1	27	21	15	9	3	28	22	16	10	4	29	23	17	11	5	30	24	18	12	6	32	21	16	11	6
C(28)	19	LB(28)	28	24	20	1	7	2	28	23	18	13	8	3	29	24	19	14	9	4	30	25	20	15	10	5	31	26	32	13	9	5
C(29)	24	LB(29)	29	26	1	16	12	8	4	31	27	23	19	15	11	7	3	30	26	22	18	14	10	6	2	29	25	21	17	32	7	4
C(30)	8	LB(30)	30	1	23	20	17	14	11	8	5	2	30	27	24	21	18	15	12	9	6	3	31	28	25	22	19	16	13	10	32	3
C(31)	22	LB(31)	1	28	26	24	22	20	18	16	14	12	10	8	6	4	2	31	29	27	25	23	21	19	17	15	13	11	9	7	5	32
C(32)	32	LB(32)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2

Basic Random sequence

permutation pattern of basic random sequence

【図１８】

q	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
$L_{32}(1)$	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	28	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2
$L_{32}(2)$	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4
$L_{32}(3)$	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10
$L_{32}(4)$	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28
$L_{32}(5)$	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20
$L_{32}(6)$	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27
$L_{32}(7)$	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17
$L_{32}(8)$	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18
$L_{32}(9)$	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21
$L_{32}(10)$	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30
$L_{32}(11)$	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26
$L_{32}(12)$	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14
$L_{32}(13)$	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9
$L_{32}(14)$	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25
$L_{32}(15)$	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11
$L_{32}(16)$	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31
$L_{32}(17)$	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29
$L_{32}(18)$	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5	23
$L_{32}(19)$	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13	5
$L_{32}(20)$	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6	13
$L_{32}(21)$	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16	6
$L_{32}(22)$	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15	16
$L_{32}(23)$	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12	15
$L_{32}(24)$	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3	12
$L_{32}(25)$	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7	3
$L_{32}(26)$	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19	7
$L_{32}(27)$	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24	19
$L_{32}(28)$	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8	24
$L_{32}(29)$	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1	8
$L_{32}(30)$	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22	1
$L_{32}(31)$	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32	22
$L_{32}(32)$	22	1	8	24	19	7	3	12	15	16	6	13	5	23	29	31	11	25	9	14	26	30	21	18	17	27	20	28	10	4	2	32